sciendo RIGA TECHNICAL UNIVERSITY *Electrical, Control and Communication Engineering*

ISSN 2255-9159 (online)
ISSN 2255-9140 (print)
2019, vol. 15, no. 1, pp. 21–29
doi: 10.2478/ecce-2019-0004
https://content.sciendo.com

Machine Learning Platform for Profiling and Forecasting at Microgrid Level

Enea Mele* (*National and Kapodistrian University of Athens, Athens, Greece*),
Charalambos Elias (*Assistant Professor, National and Kapodistrian University of Athens, Athens, Greece*),
Aphrodite Ktena (*Professor, National and Kapodistrian University of Athens, Athens, Greece*)

***Abstract* – The shift towards distributed generation and microgrids has renewed the interest in forecasting algorithms and methods, which need to take into account the advances in information, metering and control technologies in order to address the challenges of forecasting problems. Technologies such as machine learning have been proven useful for short-term electricity load forecasting, especially for microgrids, as they can also take into account several types of historical data and can adapt to changes often encountered in small-scale systems and on a short time scale. In this paper, we present a flexible and easily customized modular toolbox, called Divinus, for electricity use profiling and forecasting in microgrids. Divinus may support a variety of machine learning algorithms for forecasting and profiling that can be used independently or combined. For demonstration purposes, we have implemented Self-Organizing Maps for profiling and *k*-Neighbors for forecasting. The testing of the platform was based on electricity consumption data of the Euripus campus of the National and Kapodistrian University of Athens in Evia, Greece, from January 2010 till March 2018. The tests that have been carried out so far show that the platform can be easily customized and the algorithms examined yield high accuracy and acceptable mean errors for the case of a university campus energy profile.**

***Keywords* – Clustering algorithms; Forecasting; Machine learning algorithms.**

I. Introduction

The distributed and renewable generation is growing and new systems such as microgrids are being implemented in the distribution network. This decentralized structure poses new challenges for reliable and quality services of the electricity distribution network and entities. In order to cope with these challenges, tools for short-term load forecasting need to be developed and used in the decision-making process for optimum control, dispatching and planning as well as for demand side management, and trading in the retail electricity market. Accurate models for electric power load forecasting are essential to the operation and planning of microgrids, especially in view of the electricity market deregulation and the anticipated price fluctuations in all four markets of the target model. In light of the above, forecasting has gained an increased importance in modern day's microgrids, due to their integration in the main grid and their role in the electricity markets. Novel techniques and models are taking advantage of the advances in artificial intelligence algorithms, which allow for faster convergence, big data manipulation sets and solving more complex problems.

Algorithms where the underlying system is more or less treated as a black box and which use available data for the training of a model are usually grouped under the term machine learning. Arthur Samuel coined the term "Machine Learning" in 1959, defining it as "the ability to learn without being explicitly programmed." Machine Learning, in its most basic form, is the practice of using algorithms to parse data, learn from them, and then make a determination or prediction about something in the world.

Many of the recent advances in smart devices are the result of deep learning, a subset of machine learning that uses layers of neural networks to develop computer models that are roughly based on the structure of the human brain neurons. When machine learning systems are combined with big data, they provide for new capabilities and emerging technologies [1].

So, in contrast hard coding software routines with specific instructions to accomplish a particular task, machine learning is a way of "training" an algorithm to learn by itself. In this sense, "training" involves feeding huge amounts of data to the algorithm, allowing it to optimize itself towards performing a given task better and more quickly than expert humans [1], [2].

Machine learning is being used extensively in many problems where big amounts of data are generated, such as weather forecasting. In this work, we focus on the data relevant to electrical consumption. Long and midterm forecasting has been used for years by power companies for operation and planning purposes. They have traditionally used more conventional algorithms, such as linear regression [3] and econometric models [4] with acceptable accuracy for the task at hand.

The transition towards the smart grid and the emergence of microgrids and distributed generation and storage have dictated the restating of the load forecasting problem, setting new requirements. The amount of data generated by the new grid architecture is increasing dramatically considering the large number of intermediate and lower levels of real time metering and control introduced by new grid entities, such as smart metering, electrical vehicles, and distributed generation and storage. Algorithms that are able to classify these data and use them to train systems are required [5]. For the management of microgrids, the short-term forecasting problem needs to be solved.

In this paper, we present a modular tool, called Divinus, which is able to make hourly forecasts at microgrid level, using

* Corresponding author.
E-mail: dimitrismele@gmail.com

electrical load and, if necessary, environmental data of past years. The proposed architecture enables implementation, testing, combination and comparison of forecasting and clustering algorithms in order to obtain the optimum forecast for the problem studied. The paper is organized in four sections. Section II briefly reviews the state-of-the-art in machine learning algorithms used in load forecasting models. Section III presents the architecture of the proposed toolbox and Section IV presents and discusses preliminary results and future research directions.

II. Background and Literature Review on Machine Learning Forecasting Models

Depending on the application, electrical load forecasting algorithms are used for predicting the energy demand for the next hour (or half hour), next day, next few weeks up to the next year or periods of over a year, when investment planning is considered. Conventional forecasting algorithms have been developed taking into account the requirements and operation of the large main power grids of the conventional grid architecture. These are not appropriate for microgrids where the average and peak demand is not only several times smaller than in region-wide areas, but also its electricity consumption presents a much higher volatility [5].

In the conventional grid, which was developed based on the vertically integrated state-owned electricity companies, forecasting was used for long- and midterm forecasting for economic operation and planning. The increasing application of renewable energy sources (RES), the interconnections between countries, cross-border and wholesale energy trading have created the need for next-day or even shorter interval forecasting (see Table I). Power companies have traditionally used simple forecasting models, like linear regression [3] and econometric models [4] with satisfactory performance for the purpose they were used for. Nowadays, the increase in computing power has allowed multiple regression models to be used for very large systems [6], [7], large metropolitan areas [8], or small areas [9]. Artificial intelligence techniques have also been applied using either neural networks [10], [11], or fuzzy logic [12], [13], [14].

Lately, the efforts concentrate on unsupervised learning neural networks such as Self Organizing Maps (SOMs) [15], [16], on Neural Networks [17] or on hybrid systems that combine both Self Organizing maps (SOMs) and algorithms, such as support vector machines (SVMs) [18] or *k*-Nearest Neighbors (kNN) [5].

TABLE I
Time Horizons & Steps of Load Forecasting

	Horizon	***Step***
VSTLF	30 Min – 1 Hour	≥ 1 Minute
STLF	1 Day – 1 Week	1 Hour
MTLF	1 Year	1 Week
LTLF	10–20 Years	1 Year

In case of microgrids, however, we need to focus on short and very short-term forecasting. Short term load forecasting (STLF) refers to a time horizon of the next day or maximum of the next week by step of 1 hour, while the very short-term load forecasting (VSTLF) refers to a time horizon of next hour or half-hour by step of 1 minute. Table I summarizes the time horizons and corresponding steps in load forecasting algorithms.

Machine learning processes are so far the state-of-the-art solution in STLF and VSTLF as they can offer predictions with acceptable accuracy in a relatively short time. Depending on whether they use the labeled data, unlabeled data or both types, a machine learning process may be classified into three broad categories, namely, supervised, unsupervised and semi-supervised. Unlabeled data usually consist of samples that are artifacts created either by the monitoring system or by humans and can be readily obtained, e.g. through sensors. On the other hand, labeled data are the processed data typically made of a set of unlabeled data, where each piece of data is augmented by some sort of meaningful "tag", "label" or "class" that offers additional information or knowledge.

Another categorization of machine learning tasks is done based on the desired output. If the desired output of the model is a class, then it is a classification problem, if it is a number, then it is a regression problem and if it is a set of input groups, it is a clustering problem.

The main families of algorithms mostly used in microgrid forecasting problems are the support vector machines (SVM), *k*-Nearest Neighbors (kNN), Random Forest (RF) and artificial neural networks (ANN).

A. Support Vector Machine

Support Vector Machine (SVM) is a machine learning algorithm which can be used for both classification and regression tasks. Its formulation is based on the Structural Risk Minimization (SRM) principle, which has been shown to be superior to the traditional Empirical Risk Minimization (ERM) principle, used in conventional neural networks. SRM minimizes an upper bound on the expected risk, whereas ERM minimizes the error in the training data. It is this difference which equips SVM with a greater ability to generalize [19].

The advantage of SVM is that it is effective in high dimensional spaces and uses a subset of training points in the decision function making, it is also memory efficient. However, the algorithm does not directly provide probability estimates, because they are calculated using an expensive five- or ten- fold cross-validation [20].

Support Vector Machines models can be found in many forecasting models for both medium- and short-term forecasting or in combination with other algorithms. For instance, in [21], a method of Dragonfly Algorithm-based support vector machine (DA-SVM) was implemented to forecast the short-term load in the microgrid of an offshore oilfield group in the Bohai Sea, China. This method adopts the combination of penalty factor *C* and kernel parameters of SVM, which needs to be optimized as the position of dragonfly to find the solution. It takes the forecast accuracy calculated by SVM as the current fitness value of dragonfly and the optimal position of dragonfly obtained through iteration is considered as the

optimal combination of parameters *C* and *s* of SVM. The experimental results indicate that the DA-SVM algorithm has better global searching ability. In this case study, the root mean square errors of DA-SVA was about 1.5 % and the computation time saved was about 50 %.

B. k-Nearest Neighbors

k-Nearest Neighbors (kNN) is a supervised machine learning algorithm used for both classification and regression predictive problems.

kNN is a simple algorithm that stores all available cases and predicts the numerical target based on a similarity measure, e.g., distance, proximity, or closeness functions, as it assumes that similar data points are close to each other. It has been used as a non-parametric technique, meaning that it does not make any assumptions on the underlying data distribution. kNN methods are known as non-generalized machine learning methods, since they simply "remember" all their training data and do not have the ability to generalize [22].

The main advantage of kNN is that it is robust to noisy training data and effective if the training dataset is large. On the other hand, the user needs to determine the value of *k*, which corresponds to a positive integer, typically small, and the computation cost is high as it needs to compute the distance of each instance to all training samples.

kNN models can be found in many forecasting models. For example, a kNN algorithm was used to construct a hybrid model comprised of a Wavelet Denoising-Extreme Learning Machine (ELM) and a kNN Regression in order to have a half-hour electricity load forecast, in New South Wales [23]. The training was done making use of 2 832 training load data from 1 January 2017 00:30 to 1 March 2017 00:00 and 2 158 testing load data from 1 March 2017 00:30 to 14 April 2017 23:30. In [24], kNN models were used for day ahead load prediction using only limited temperature data as inputs, namely, the minimum and maximum daily temperatures.

C. Random Forests

The Random Forest (RF) technique is used in many forecasting models due to its simplicity and the fact that it can be used for both classification and regression tasks. RF is a supervised learning algorithm which builds multiple decision trees and merges them together to get a more accurate and stable prediction. More specifically, it constructs a multitude of decision trees during training and yields the class that is the mode of the classes (classification) or mean prediction of the individual trees (regression). RF corrects the decision trees habit of over fitting their training set because no single tree can learn from all of the instances and explanatory variables. No single tree can memorize all of the noise in the representation [25].

Its default hyperparameters often produce a good prediction result, which makes RF a very handy and easy to use algorithm. Its main limitation is the number of trees which it needs to yield these results. As the trees grow, the algorithm speed slows down and it becomes ineffective for real-time predictions.

RF has been widely used in forecasting models. In [26], it was proposed to improve the accuracy in a STLF problem by combining it with a feature selection method based on the generalized minimum redundancy and maximum relevance. Another paper proposes using RF models for short-term electric load forecasting, making use of an ensemble learning method that generates many regression trees and aggregates their results [27].

D. Artificial Neural Networks

Neural Networks (ANN) are a class of models within the general machine learning literature and also a specific set of algorithms that have revolutionized machine learning. The operating principle of ANNs is based on the functions of the human brain neurons; they provide powerful tools for modelling, especially when the underlying data relationship is unknown. Moreover, they can identify and learn correlated patterns between input data sets and corresponding target values [11], [15], [28]. They have been successfully applied in a variety of scientific fields, such as mathematics, engineering, medicine, economics, meteorology, psychology, neurology and many other.

They have also attracted a lot of criticism on the grounds of (a) their "black-box" treatment of the system being modeled, which does not provide any insight regarding why and how the result was obtained; (b) the absence of specific rules for the determination of the network structure and the trial–and–error approach which reduces their trustworthiness as forecasting tools; (c) the fact that they are computationally intensive and therefore hardware dependent as they may require processors with parallel processing power, depending to their structure; (d) the fact that the network's performance is evaluated against specific preset error values, which is the criterion for ending the training, even though this error value does not necessarily yield the optimum results [11], [15], [28].

The reason why they have been applied in so many scientific fields is partly explained by the following four factors [11], [15], [28]:

- There is an abundance of data generated in modern measurement systems, which can be utilized by the fairly large database required by ANNs where known inputs are compared with the corresponding outputs in order to "educate" the network.
- The iterative comparison of the output value with the actual one and the amendment of the weights in accordance with the "education rule" and the calculated error decreasing with the increasing number of repetitions makes them an attractive option for systems where time series data are generated.

There are dozens of forecasting models that all these years have been implemented on ANNs.

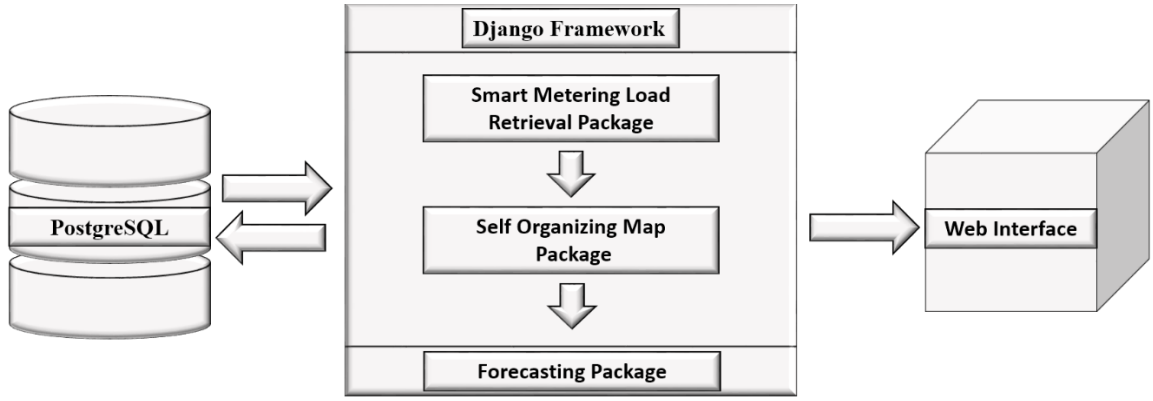

Fig. 1. Divinus architecture.

For example, the model proposed in [29] for STLF in microgrids is based on a three-stage architecture, which starts with pattern recognition by a Self-Organizing Map (SOM), a clustering of the previous partition via a *k*-means algorithm, and finally demands forecasting for each cluster with a MultiLayer Perceptron Model, which is a class of feedforward ANN. Validation was performed with the data from a microgrid-sized environment provided by the Spanish company Iberdrola. In [30], a Self-Recurrent Wavelet Neural Network (SRWNN) is used as a forecast engine in a microgrid using the Levenberg-Marquardt (LM) learning algorithm to train the SRWNN. In order to demonstrate the efficiency of the proposed method, it was tested against actual hourly load data of an educational building supplied by a microgrid.

III. Divinus Architecture

The motivation behind developing the Divinus platform was to create a user-friendly tool able to put to test different existing or new forecasting and clustering algorithms working separately or combined.

Divinus is a modular platform which allows to easily add, remove or modify various parts of it. It has a user-friendly interface whose only requirement is the proper formatting of the dataset to be used as input to the algorithms.

The architecture of Divinus consists of several interconnected well-defined components where each one interacts directly with the other as it is depicted in Fig. 1. The tools used are a database where all information is stored, an appropriate programming language and a website hosting and displaying the results.

The database allows storing data dynamically. The decision for the choice of the appropriate technology for this task was based on two criteria. The first one was the type of data to be processed. Our data consists of dates and time intervals associated with load as well as environmental data. This requires the use of a relational database and therefore only Structured Query Language (SQL) databases were acceptable. Second, for conceptual as well as cost considerations, we decided to use open source technologies. This narrowed down our choices to open-source relational databases. PostgreSQL was chosen as the most advanced open-source database available today [31].

Next we had to decide on the programming language to build Divinus and which had to be compatible with machine learning. Python was chosen as one of the most popular programming languages for machine learning and data mining. It enjoys a large number of useful add-on libraries and frameworks that are developed by a constantly increasing community. One of the biggest advantages of Python is Django, the open source high-level Python Web framework [32] in which the core functionalities of our platform are based.

The website hosting platform has been developed with the use of HTML, CSS and Javascript and is the point where all the data generated through the clustering and forecasting algorithms are collected and displayed. The website is connected, through Django, with the database from where all data are retrieved in Json format and are graphically presented. In such a way, the end user can interface in a friendly and efficient way with the platform and preview the profiles that were created or the loads that were forecasted for the days ahead.

Next, we present the functional components of the platform, which enable the collection and organization of the data that will be saved in the database, the profiling and clustering of the collected data and the load forecasting based on the extracted profiles.

A. Data Collection

Data collection and management are one of the most important aspects of such platforms. The quality, granularity and format of data must be such that it should yield high accuracy results. Since the platform uses actual data, it is important to determine that the format and structure of the data input to the algorithms are appropriate. Divinus is designed to accept electricity consumption data, as well as environmental data, such as temperature, sun radiation, humidity, wind speed and pressure, and to specify the time interval in which the predictions will take place. However, in the current phase of development, we present test results concerning its forecasting potential, using the data which have previously been clustered using SOM. For this, we have used only electricity consumption data under the assumption that the clustered data have already incorporated the influence of the environmental conditions.

The data used for testing correspond to the electricity consumption of the Euripus campus of the National and Kapodistrian University of Athens (NKUA). The data were retrieved from the telemetering service of HEDNO, the Greek DSO of the Public Power Corporation (PPC), and consist of hourly measurements for both active and reactive loads from January 2010 to February 2018. Hourly measurements were used because this is the data granularity provided by HEDNO.

First, the database was connected to the Django framework so that the tables created in the database could be used by the framework to retrieve or store data through PostgreSQL commands [31].

The next thing was to retrieve the data from the HEDNO telemetering service. As there was no API available to download the required data, an interface subroutine was developed through which Divinus entered the telemetering service at the beginning of each day and downloaded the data of the previous day in Excel format. Another subroutine then processed them and stored them in the Divinus database. If another metering service were to be accessed, this interface module would have to be adjusted to the specifics of the metering service. Using a python library called pandas, small changes and corrections were made to the data without damaging their formatting. These changes may involve converting one type of number into another or converting a file type to another in order to be easily editable.

Next, the data were imported in the database using postgreSQL commands [31].

B. Clustering Algorithm – Self-Organizing Maps

After importing and organizing the measurement data, we proceed with data clustering.

The technology chosen for this goal is the Self Organizing Map (SOM) which falls under the category of unsupervised learning algorithms. SOM is a type of Artificial Neural Network able to convert complex, nonlinear statistical relationships between high-dimensional data items into simple geometric relationships on a low-dimensional display [33].

However, data pre-processing is required to take place before implementing the SOM algorithm. The data that were loaded in our system contained hourly values, which means that they contained a timestamp and the hourly consumption. This format, however, was not the desired one because although they could be clustered by the SOM, the clusters that would be created would not be useful for the forecasting algorithm following next. Therefore, the data were reorganized in an appropriate format.

The SOM implementation used is the minisom [32] algorithm as retrieved from the Python library, which is a minimalistic and Numpy based implementation of SOMs.

First, we split the data into the data to be clustered and the data used to generate the clusters. Next, we specified the size of SOM and therefore the number of clusters needed for our type of data. This is application specific. Taking into consideration the use of the building as a university building hosting offices, classrooms and laboratories, we used four clusters. In particular, based on the common features that exist in various consumption profiles, one cluster includes the weekdays, the second cluster includes the weekends, the third cluster includes the holidays and the fourth includes consumption patterns that do not fall under any of the previous categories.

The next step is the SOM training, which yields the row and column of the cluster where each data point belongs. After the training, our dataset has two additional pieces of information: the SOM row and the SOM column that act as identifiers of the cluster to which each data point belongs. As soon as these two fields are added in the reformatted dataset, a loop process passes these identifiers to the original dataset that is the one on which the pre-processing was performed. The data are now eligible to be used by the load forecasting algorithm.

C. Machine Learning Forecasting Algorithm – k-Neighbors

Typical STLF algorithms rely on the use of big amounts of past data to minimize the prediction error. In this work, we use the Divinus platform to investigate whether forecasting can be used with the previously clustered data and yield the same or better accuracy which would justify inserting one more level of data processing.

The Divinus forecasting is a three-step process. The first stage of this process is to retrieve all the data needed based on the hourly consumption we want to forecast. This is done by retrieving the corresponding hourly consumption for the previous years. By retrieving the row and the column of the cluster in which they belong we are able to retrieve all the data contained in it. The second stage is to use these data for training and testing of the forecasting algorithm and the final stage is to perform the forecasting of the days that we wanted to predict.

The forecasting algorithm used is the *k*-neighbors algorithm. The principle behind nearest neighbor methods is to find a predefined number of training samples closest in Euclidean distance to the new point, and predict the future hourly active loads. The number of neighbors, from which it will retrieve samples, is a user-defined value.

In our case, the data have previously been clustered with SOM, so we need an algorithm such as *k*-neighbours to forecast future loads based on the corresponding past values that have common features. In our forecast implementation, the nearest neighbors used were $k = 3$.

IV. Results and Discussion

According to our proposed methodology, the first step that should be completed in order to start the forecasting process is the creation of the clusters. Through SOM we create clusters out of the past data. The parameters that are used for SOM adjustments are the following:

a) *x*: *x* dimension of the SOM;

b) *y*: *y* dimension of the SOM;

c) *input_len*: Number of the elements of the vectors in input;

d) *neighborhood_function*: Function that weights the neighborhood of a position in the map. Possible values are the 'gaussian' or the 'mexican_hat';

e) *sigma*: Spread of the neighborhood function, needs to be adequate to the dimensions of the map;

f) *decay_function*: Function that reduces learning rate and *sigma* at each iteration [31].

TABLE II
SOM PARAMETERS USED IN DIVINUS [25]

SOM Parameters	
x	2
y	2
Input length	25
Neighborhood function	Gaussian
Sigma	1.0
Decay function	None

The parameters selected for SOM implementation were based on the clustering requirements posed by our data. The *x* and *y* dimensions were set to 2 in order to obtain four clusters (1-1, 1-2, 2-1, 2-2): based on the available historical data, four main patterns could be identified for the campus electricity consumption, two for the fall and spring semesters, one for the holidays, such as Christmas and Easter, and one for the days that did not fall under any of the previous categories, e.g. a day where a power failure occurred. The input length was set to 25 for the day of the year and 24 hours of the day. The next parameter was sigma, which is the radius of neighbors in the SOM landscape. The default value of 1.0 was kept since trials with other values did not yield any significantly different results.

Each cluster contains days that have similar consumption profiles. Figs. 2 and 3 show the results of the clustering for the active loads of 2017. In Fig. 2, the weekdays of 2017 are displayed, where some of them are weekdays with higher consumption than others, while others – with lower. For instance, the summer months, although they have the same pattern, have a lower consumption. Accordingly, Fig. 3 displays the weekends of 2017 following the same logic. Moreover, the clusters are coded as a two-dimensional matrix due to the fact that SOM translates all the data it clusters in two dimensions *x* and *y*, and therefore when we want to show a specific cluster, we just need to specify the *x* and *y*. In the figures, for readability purposes, we only show selected profiles. The platform user is able to preview the clusters he/she wants for the year of choice.

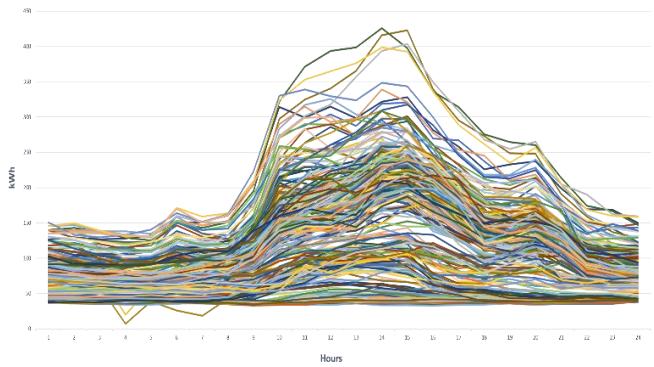

Fig. 2. SOM Cluster [0,0] containing 255 days of 2017.

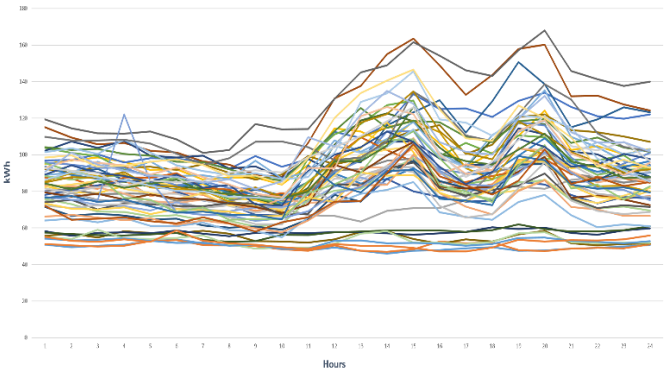

Fig. 3. SOM Cluster [1, 1] containing 56 days of 2017.

As soon as the clusters are created, we are able to proceed with load forecasting. The hourly forecast test results were conducted for:

- the next month;
- the next year.

It should be noted that in the following forecasting results, holidays have been removed and only the data on weekdays and weekends have been used. The main criterion of evaluation is MAPE (Mean Absolute Percentage Error) which is calculated by (1) and is applied to *N* data points representing hours of the month or year:

$$MAPE = \frac{1}{N} \cdot \sum_{i=1}^{N} \left| \frac{y_i - \hat{y}_i}{y_i} \right|, \quad (1)$$

where y_i is the real load and $\hat{y}_i$ is the predicted load.

A. One Month Ahead Hourly Prediction

Overall, the forecasting was successful since the predicted hourly load is very close to the actual one for both February and March with a MAPE of 12.62 % for February and 7.84 % for March respectively, as shown in Table III and Table IV. Given that the load curves in question have high volatility and the system that we are studying is a low-consumption system compared to the consumption of a real microgrid, the resulting mean errors are considered small.

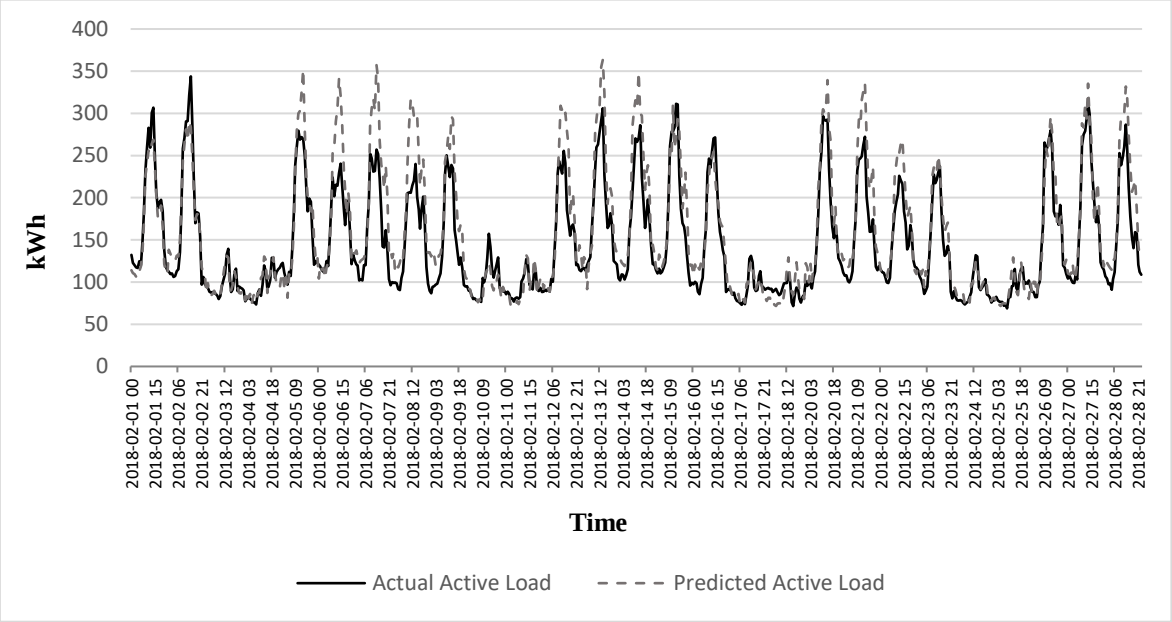

Fig. 4. Hourly prediction for February 2018: from 01/02/2018 to 28/02/2018.

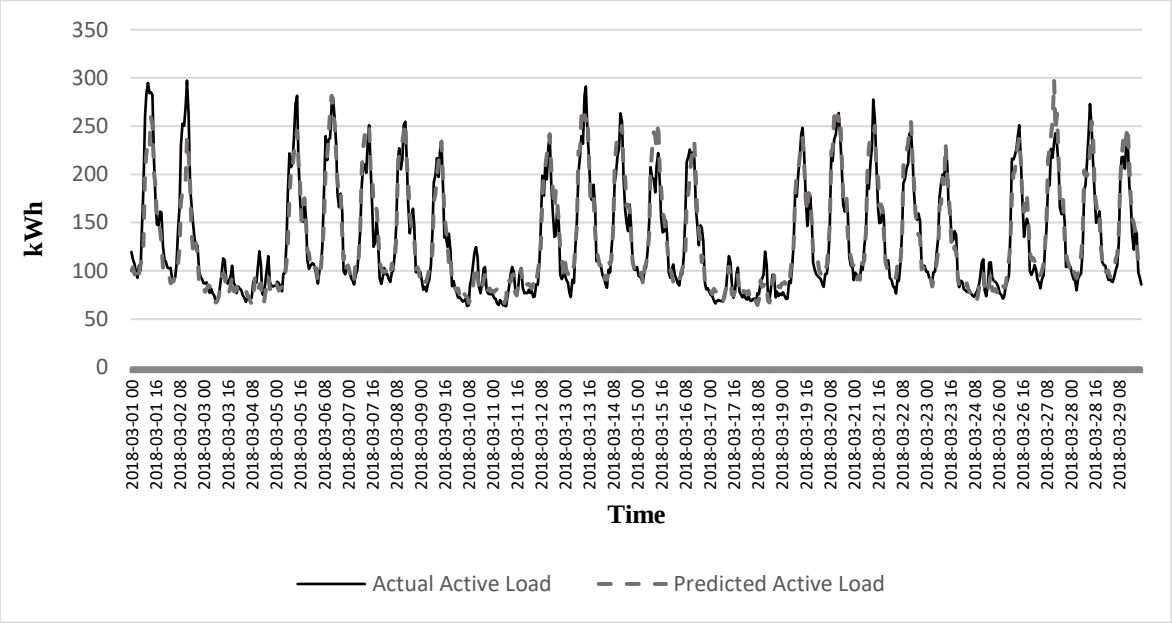

Fig. 5. Monthly prediction for March 2018 from 01/03/2018 to 31/03/2018.

Figs. 4 and 5 show the respective graphical representations of the actual and forecasted load curves for February 2018 and March 2018.

TABLE III
FORECASTING ERROR FOR FEBRUARY 2018

February 2018	
MAPE	12.62 %
Maximum Absolute Percentage Error	55.43 %

TABLE IV
FORECASTING ERRORS FOR MARCH 2018

March 2018	
MAPE	7.84 %
Maximum Absolute Percentage Error	34.44 %

B. One Year Ahead

The second test scenario was the hourly forecast for a year. Having data from 2010 up to 2016, we used the toolbox to forecast hourly loads of 2017. The resulting MAPE was

15.96 %. The MAPE for every month and the average for the whole year 2017 is shown in Table V.

Another similar test used data from 2010 to 2012 to predict the hourly loads of 2013. In this case, the training set included fewer years. The MAPE for every month and for the whole year of 2013 is shown in Table VI.

TABLE V
FORECASTING ERROR FOR 2017

2017 Predictions	MAPE
January	16.10 %
February	20.57 %
March	13.03 %
April	17.43 %
May	11.27 %
June	21.48 %
July	29.20 %
August	13.78 %
September	15.55 %
October	10.73 %
November	11.75 %
December	13.66 %
2017 MAPE	**15.96 %**
Maximum Absolute Percentage Error	**164.29 %**

TABLE VI
FORECASTING ERRORS FOR 2013

2013 Predictions	MAPE
January	13.71 %
February	35.96 %
March	14.78 %
April	14.97 %
May	10.13 %
June	14.33 %
July	34.87 %
August	35.52 %
September	26.68 %
October	16.35 %
November	16.24 %
December	17.13 %
2013 MAPE	**20.26 %**
Maximum Absolute Percentage Error	**269.52 %**

The results are acceptable for both cases shown in spite of the average MAPE of 2013 being larger. This may be due to the fact that only 1/3 of the data was used for the training. Also, the months that have a higher error include summer months that generally present a higher volatility.

The cases shown are not intended to test the performance of two well-known algorithms, namely SOM and kNN. They are rather used to illustrate the operation of the Divinus platform. More cases need to be studied and further tests need to be carried out with and without clustering using several forecasting techniques and the results should be compared with the published data. However, certain conclusions may be drawn even at this stage.

Since the electricity demand in microgrids is characterized by a much higher volatility than the one in the main power grid, it would not be useful to compare these preliminary results against forecasting results for the main grid. Here we report on the comparison against results which combine SOMs with Neural Networks [15] for 20 days ahead load forecasting in a microgrid, using the data of 4 months from 02/12/2010 to 07/05/2011 for training, testing, and validation with percentages 60 %, 30 %, and 10 %, respectively. The MAPE errors reported for 1 hour, 12 hours, 1 day and 2 days were 12.851 %, 13.712 %, 13.810 % and 14.495 %, respectively. The error increases when a longer step-ahead prediction was considered. Divinus performed hourly forecasts for the next year and the MAPE errors were found to be 15.96 % and 20.26 % for 2017 and 2013, using training data of 5 and 2 years, respectively.

Ongoing research involves the use of Divinus for the study of the performance of other algorithms for clustering and forecasting, used independently or in conjunction with each other, and the design of a subroutine that will enable forecasting for holidays, testing for lower forecasting time intervals and the optimal use of the functionality for environmental data. Also, electricity use forecasting studies will be carried out using various algorithms whose best results will be combined and integrated into one prediction.

V. CONCLUSION

This article presents a modular platform created for electricity consumption data clustering and forecasting in microgrids. The platform is built using open source tools, it is modular and independent of the algorithms it supports. The platform is user friendly and allows the user to select the data set and procedure for forecasting, as well as preview clustering and forecasting results in an interactive manner. In addition, due to its high modularity, it can be used as a test workbench through which the interaction between different algorithms can be measured.

For the preliminary test results presented here, clustering was performed using Self-Organizing Maps and forecasting was performed with the use of the *k*-neighbors algorithm. The data from a university building were first clustered and then used for load forecasting. The proposed approach yielded very good accuracy for hourly forecasts of the next month and the next year.

REFERENCES

[1] T. V. Ark, "Ask About AI - The Future of Work and Learning," Getting Smart Staff, 2017.

[2] E. Koblentz, "How to implement AI and machine learning," TechRepublic, 2016.

[3] Z. Mohamed and P. Bodger, "Forecasting electricity consumption in New Zealand using economic and demographic variables," *Energy*, vol. 30, no. 10, pp. 1833–1843, 2005. https://doi.org/10.1016/j.energy.2004.08.012

[4] M. Yang and X. Yu, "China's rural electricity market – a quantitative analysis," *Energy*, vol. 29, no. 7, pp. 961–977, 2004. https://doi.org/10.1016/j.energy.2003.12.002

[5] E. Mele, A. Ktena and C. Elias, "Electricity use profiling and forecasting at microgrid level," in *RTUCON 2018*, Riga, Latvia, 2018. https://doi.org/10.1109/RTUCON.2018.8659866

[6] T. Haida and S. Muto, "Regression based peak load forecasting using a transformation technique," *IEEE Trans. Power Syst.*, vol. 9, no. 4, pp. 1788–1794, Nov. 1994. https://doi.org/10.1109/59.331433

[7] S. Mirasgedis, Y. Safaridis, E. Georgopoulou, D. P. Lalas, M. Moschovits, F. Karagiannis and D. Papakonstantinou, "Models for mid-term electricity demand forecasting incorporating weather influences," *Energy*, vol. 31, no. 2-3, pp. 208–227, 2006. https://doi.org/10.1016/j.energy.2005.02.016

[8] H. L. Willis and J. E. D. Northcote-green, "Comparison tests of fourteen distribution load forecasting methods," *IEEE Trans. Power App. Syst.*, vol. 103, no. 6, pp. 1190–1197, 1984. https://doi.org/10.1109/TPAS.1984.318448

[9] H. L. Willis, R. W. Powell, and D. L. Wall, "Load transfer coupling regression curve fitting for distribution load forecasting," *IEEE Trans Power App. Syst.*, vol. 103, no. 5, pp. 1070–1076, 1984. https://doi.org/10.1109/TPAS.1984.318713

[10] E. Doveh, P. Feigin, D. Greig and L. Hyams, "Experience with FNN models for medium term power demand predictions," *IEEE Trans. Power Syst.*, vol. 14, no. 2, pp. 538–546, May 2002. https://doi.org/10.1109/59.761878

[11] G. J. Tsekouras, N. D. Hatziargyriou and E. N. Dialynas, "An optimized adaptive neural network for annual midterm energy forecasting," *IEEE Trans. Power Syst.*, vol. 21, no. 1, pp. 385–391, Feb. 2006. https://doi.org/10.1109/TPWRS.2005.860926

[12] M. Y. Chow, J. Zhu and H. Tram, "Application of fuzzy multi-objective decision making in spatial load forecasting," *IEEE Trans. Power Syst.*, vol. 13, no. 3, pp. 1185–1190, Aug. 1998. https://doi.org/10.1109/59.709118

[13] C. N. Elias and N. D. Hatziargyriou, "An Annual Midterm Energy Forecasting Model Using Fuzzy Logic," *IEEE Transactions On Power Systems*, vol. 24, no. 1, pp. 469–478, Feb. 2009. https://doi.org/10.1109/TPWRS.2008.2009490

[14] G. J. Chen, K. K. Li, T. S. Chung, H. B. Sun and G. Q. Tang, "Application of an innovative combined forecasting method in power system load forecasting," *Elect. Power Syst. Res.*, vol. 59, no. 2, pp. 131–137, 2001. https://doi.org/10.1016/S0378-7796(01)00137-7

[15] J. Llanos, D. Sáez, R. Palma-Behnke, A. Núñez and G. Jiménez-Estévez, "Load Profile Generator and Load Forecasting for a Renewable Based Microgrid Using Self Organizing Maps and Neural Networks," in *The 2012 International Joint Conference on Neural Networks (IJCNN)*, 2012. https://doi.org/10.1109/IJCNN.2012.6252648

[16] Z. H. Bohari, H. Azemy, M. N. M. Nasir, M. F. Baharom, M. F. Sulaima and M. H. Jali, "Reliable Short Term Load Forecasting Using Self Organizing Map (SOM) In Deregulated Electricity Market," *Journal of Theoretical and Applied Information Technology*, vol. 79, no. 3, pp. 389–394, Sep. 2015.

[17] O. A. S. Carpinteiro and A. J. R. Reis, "A Hierarchical Self-Organizing Map Model In Short-Term Load Forecasting," *Journal of Intelligent and Robotic Systems*, vol. 31, no. 1-3, pp. 105–113, May 2001.

[18] J. Che, J. Wang and G. Wang, "An adaptive fuzzy combination model based on self-organizing map and support," *Energy*, vol. 37, no. 1, pp. 657–664, Jan. 2012. https://doi.org/10.1016/j.energy.2011.10.034

[19] V. Jakkula, "Tutorial on Support Vector Machine (SVM)", School of EECS, Washington State University, 2012.

[20] "1.4. Support Vector Machines," scikit-learn, [Online]. Available: https://scikit-learn.org/stable/modules/svm.html. [Accessed 3 May 2019].

[21] A. Zhang, P. Zhang and Y. Feng, "Short-term load forecasting for microgrids based on DA-SVM," *COMPEL - The International Journal for Computation and Mathematics in Electrical and Electronic Engineering*, vol. 38, no. 1, pp. 68–80, 2019. https://doi.org/10.1108/COMPEL-05-2018-0221

[22] "scikit-learn," scikit-learn developers, [Online]. Available: http://scikit-learn.org/stable/modules/neighbors.html. [Accessed May 26 2018].

[23] W. Li, D. Kong and J. Wu, "A Novel Hybrid Model Based on Extreme Learning Machine, k-Nearest Neighbor Regression and Wavelet Denoising Applied to Short-Term Electric Load Forecasting," *Energies 2017*, vol. 10, no. 5, 2017. https://doi.org/10.3390/en10050694

[24] R. Zhang, Y. Xu, Z. Y. Dong, W. Kong and K. P. Wong, "A composite k-nearest neighbor model for day-ahead load forecasting with limited temperature forecasts," *2016 IEEE Power and Energy Society General Meeting (PESGM)*, Jul. 2016. https://doi.org/10.1109/PESGM.2016.7741097

[25] G. Hackeling, "Mastering Machine Learning with scikit-learn", Birmingham: Packt Publishing, October 2014.

[26] N. Huang, Z. Hu, G. Cai and D. Yang, "Short Term Electrical Load Forecasting Using Mutual Information Based Feature Selection with Generalized Minimum-Redundancy and Maximum-Relevance Criteria," *Entropy 2016*, vol. 18, no. 9, 2016. https://doi.org/10.3390/e18090330

[27] G. Dudek, "Short-Term Load Forecasting using Random Forests, Department of Electrical," in Filev D. et al. (eds) Intelligent Systems' 2014. *Advances in Intelligent Systems and Computing*, vol. 323. Springer, Cham. 2015. https://doi.org/10.1007/978-3-319-11310-4_71

[28] S. A. Kalogirou, "Artificial neural networks in energy applications in buildings," *International Journal of Low Carbon Technologies*, vol. 1, no. 3, pp. 201–216, Jul. 2006. https://doi.org/10.1093/ijlct/1.3.201

[29] L. Hernández, C. Baladrón, J. M. Aguiar, B. Carro, A. Sánchez-Esguevillas and J. Lloret, "Artificial Neural Networks for Short-Term Load Forecasting in Microgrids Enviroment," *Energy*, vol. 75, pp. 252–264, Oct. 2014. https://doi.org/10.1016/j.energy.2014.07.065

[30] H. Chitsaz, H. Shaker, H. Zareipour, D. Wood and N. Amjady, "Short-term Electricity Load Forecasting of Buildings in Microgrids," *Energy and Buildings*, vol. 99, pp. 50–60, Jul. 2015.

[31] The PostgreSQL Global Development Group, "PostgreSQL," The PostgreSQL Global Development Group, [Online]. Available: https://www.postgresql.org/about/. [Accessed 20 02 2018].

[32] Django Software Foundation, "django," Django Software Foundation, [Online]. Available: https://www.djangoproject.com/start/overview/. [Accessed 26 May 2018].

[33] G. Vettigli, "MiniSom: minimalistic and NumPy-based implementation of the Self Organizing Map," 15 September 2013. [Online]. Available: whttps://github.com/JustGlowing/minisom. [Accessed 24 May 2018].

Enea Mele holds a B.Sc. degree in Electrical Engineering (2015) and an M.Sc. degree in intelligent management of renewable energy sources (2017).
He is currently working at Wirecard as a Senior Automation Tester and engaged in research projects with the Energy Systems Laboratory of the National and Kapodistrian University of Athens, Greece. He has specialized in gaming technologies and previously worked at SAICON, a company dedicated to game development and gamified mobile applications and Intralot, a company dedicated to integrated game content, sports betting management and interactive gambling services.
His research interests include RES microgrids, load and energy forecasting methods, machine learning and AI algorithms.
Contact details: National & Kapodistrian University of Athens, Euripus Campus, Psachna 34400, Greece.
E-mail: dimitrismele@gmail.com

Charalambos N. Elias is an Assistant Professor at the General Department, of National and Kapodistrian University of Athens, Greece, since 2017. He received the diploma in electrical and mechanical engineering from the Electrical Engineering Department at Aristotle University, Thessaloniki, Greece, in 1996, and the M.Sc. and Ph.D. degrees from the National Technical University of Athens (NTUA), Athens, Greece, in 2002 and 2012, respectively.
His research interests include power system analysis, load and energy forecasting methods, renewable energy sources and microgrids, fuzzy expert systems.
Member of the Technical Chamber of Greece.
Contact details: National & Kapodistrian University of Athens, Euripus Campus, Psachna 34400, Greece.
E-mail: cilias@uoa.gr

Aphrodite Ktena holds Ph.D. & M.Sc. degrees (1993) in Electrical & Computer Engineering from Carnegie Mellon University, USA and a B.Sc. degree (1989) in Electrical Engineering from the University of Bridgeport, USA.
She is a tenured Professor and member of the Energy Systems Laboratory of the National and Kapodistrian University of Athens, Greece. Her research interests include RES microgrids, sensor development and measurement technology, hysteresis modelling, system optimization, magnetic non-destructive testing.
Member of the Technical Chamber of Greece, CIGRE and IEEE.
Contact details: National & Kapodistrian University of Athens, Euripus Campus, Psachna 34400, Greece.
E-mail: apktena@uoa.gr
ORCID iD: https://orcid.org/0000-0003-1350-2408